Device for Secure Access to Digital Media Contents, Virtual Multi-Interface Driver and System for Secure Access to Digital Media Contents

The present invention relates to a device for secure access to digital media contents. The invention further relates to a virtual multi-interface driver and to a system for secure access to digital media contents.

Secured data storage has become a new application for digital media. All digital media do not have in-built security. Hence in order to store bulk data in a secured fashion it is required to add some external security mechanism. Smart card protection is one ideal candidate for such a mechanism as it is one of the most proven technologies for security products.

Media containing embedded smart card controllers have reached the market. Hence it has become a necessity for the device to support smart card commands. But most of the digital media readers available in the market are single interface devices that are mass storage compliant. They cannot directly support the new media with embedded smart card controllers due to their architectural limitation.

In order to prevent unauthorized access to digital media contents and to overcome the above-mentioned architectural limitation of single interface devices, the invention provides device for secure access to digital media contents as recited in claim 1, a virtual multi-interface driver as recited in claim 18 and a system for secure access to digital media contents as recited in claim 24. Expedient and advantageous embodiments of the invention are recited in the subclaims.

Further details and advantages of the invention become apparent from the following description of several prior art systems for access to digital media contents and of a preferred embodiment of the invention. The description makes reference to the accompanying drawings, in which:

Figure 1 shows a prior art system including a single interface USB device;

Figure 2 shows a prior art system including a composite device;

Figure 3 shows a prior art system according to the core USB framework;

Figure 4 shows a prior art system according to an extended USB device framework;

Figure 5 shows a schematic electrical diagram of a further prior art system;

Figure 6 shows a schematic electrical diagram of a system according to a preferred embodiment of the invention;

Figure 7 shows possible application scenarios for the virtual multi-interface driver according to the invention;

Figure 8 shows a logical connection diagram of the system according to the preferred embodiment of the invention;

Figure 9 shows the software architecture of the system according to the preferred embodiment of the invention; and

Figure 10 is a command flow diagram for the device according to the preferred embodiment of the invention.

Figure 1 illustrates a prior art system according to the MSDN Library (under the topic: Windows Driver Stack for Windows XP and LATER, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/buses/hh/buses/usbsystem_6ofb.asp). The device shown in Figure 1 is a single interface USB device that has either a mass storage interface (left part of Figure 1) or a smart card interface (right part of Figure 1). The driver loaded for the device is provided by Microsoft Windows OS. Only the functionality of one of the interfaces can be achieved at an instance, depending on whether it is a digital media reader or a smart card reader. This architecture is incapable of supporting a second device

function (e.g. a smart card reader in addition to a digital media reader) as the device only has a single physical interface.

Figure 2 shows a further prior art system according to the MSDN Library (under the topic: Selecting the Configuration for a Multiple-Interface (Composite) USB Device, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/buses/hh/buses/usb-config_6xev.asp). The device shown in Figure 2 is a composite device which has two interfaces defined in its configuration descriptor. One interface is confined to mass storage class and the other interface is confined to the class of smart cards. Both interfaces do physically exist in the device itself (although the device only comprises a single connector). Microsoft Windows OS provided drivers get loaded separately for each interface. The functionalities of both mass storage interface and smart card interface are available. This type of architecture has a limitation in that, for achieving the functionality and the intelligence of a multiple interface device, it is a must that the device itself contains multiple interfaces. Devices with a single physical interface cannot benefit from this architecture. Also, it requires both digital media and the smart card to be present in the reader for communicating with their respective interfaces. Further, this architecture cannot support a single digital medium with a smart card controller embedded within it.

The prior art system illustrated in Figures 3 also is a system according to MSDN Library (under the topic: Windows Driver Stack for Windows XP and LATER, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/buses/hh/buses/usbsystem_6ofb.asp). As can be seen in Figure 3 the base configuration model assumed by the core USB framework imposes a one-to-one association between an interface and a device function. System software is designed to the intent of the core specification and assumes one driver per function and one interface.

The prior art system shown in Figure 4 is in accordance with an extended USB device framework (see USB Engineering Change Notice, Title: Interface Association Descriptors, applying to Universal Serial Bus Specification, Revision

2.0), defining a new standard descriptor and interface descriptor that allows a device to describe which interfaces are associated with the same device function. This allows the operating system to bind all of the appropriate interfaces to the same driver instance. Figure 4 shows that device class specifications have defined device functions that use multiple interfaces. A functional driver gets loaded for a device which contains two interfaces (0 and 1), i.e. the model uses one functional driver per function, but binds multiple interfaces to the same driver instance.

Figure 5 shows a prior art system that provides a certain level of security to digital media contents. The system is a hub-based solution that contains both a digital media reader and a smart card reader. The two readers are internally connected to a USB hub that is connected, in turn, to the USB port. Each of the readers has an individual host interface. One of the shortcomings of this solution is that the host computer requires two interfaces, i.e. two USB ports. Another drawback is that the data being sent to the digital media through the digital media reader can be tapped easily at the host interface points. Hence, security is compromised.

Figure 6 shows a system for secure access to digital media contents according to a preferred embodiment of the invention. The system includes two major components: a device according to the invention, hereinafter referred to as "secure digital media reader", which is connected to a host (a PC, for example), and a "virtual multi-interface driver" according to the invention, which will be described in detail later.

The secure digital media reader includes an access means for accessing digital media contents from a data source, hereinafter referred to as "digital media reader", and a reader for authenticating a user, in particular a smart card reader. The two readers are located in a single external housing. The device can be either accommodated inside the host or be an external unit remote from the host. The digital media reader and the smart card reader may be two independent units or a single integrated unit, i.e. each reader may have its own processor unit.

The digital media reader is the device through which the digital media contents are accessed. The digital media can be interfaced to the digital media reader through any suitable standard interface such as Compact Flash (CF), Smart Media (SM), Secure Digital, Picture Card (xD), Multimedia Card (MMC), etc. (see IF 1 in Figure 6). The digital media reader can be a module, a system-on-chip (SOC) or a single chip system.

The smart card reader communicates with a smart card, which may be embedded inside the smart card reader. On the smart card electronic key information (a digital key) required to access the digital media contents is stored. The smart card can be interfaced to the smart card reader through any suitable standard interface such as ISO 7816, I2C, Contactless Smart Card Interface, etc. (see IF 2 in Figure 6). The smart card reader can be a module, a system-on-chip (SOC) or a single chip system.

There is an internal communication channel between the smart card reader and the digital media reader. This type of communication is used to guarantee a secure transfer of the digital key from the smart card reader to the digital media reader. It is thus ensured that the digital key is not externally visible for any snooping. The communication channel may also be used to transfer a PIN code to the digital media reader for additional security. In other words, the internal communication channel between the smart card reader and the digital media reader is used to protect the secure data communication within the secure digital media reader to provide a very high level of security.

As can be seen in Figure 6, only a single data channel between the secure digital media reader and the host is provided, using an electrical industry standard interface, which may be an interface designed for wireless data communication. Suitable interface standards include USB, SCSI, Firewire, WiFi, Bluetooth, HyperLAN.

The digital media contents in the media reader are available to the host only when the correct smart card has been inserted and authenticated. The digital key is

not compromised since the key is not transferred through an open channel. Thus, a user cannot access the digital media contents as he does with unsecured digital media. A smart card with a proper digital key stored thereon (optionally in combination with a PIN code entered by means of a PIN pad provided on the device or a keyboard of the host) has to be used to access the digital media content. A mismatch in smart card (or PIN code) will result in denial of authentication to access the media. Thus, only the owner can access the digital media contents.

The invention makes use of a common software layer referred to as "virtual multi-interface driver for secure media". The virtual multi-interface driver transforms the secure digital media reader into one or more of the following:

- a standard mass storage compliant reader, after proper authentication;

- a standard CCID/PCSC compliant smart card reader;

- a secure digital media reader, which shall allow access to digital media contents only on authentication with a smart card. The digital media contents may be partitioned with a secure and an unsecured portion. In this case the secure portion can be accessed only after authentication, while the unsecured portion is always available for access by the user.

The concept of the virtual multi-interface driver according to the invention will now be described. In general, a driver is a software component that acts as an interface between a device and an application software. A multi-interface USB device or a composite USB device has more than one USB interface, e.g. a mass storage class interface and a CCID interface. In other words, a composite device with a mass storage interface and a CCID interface can function both as a mass storage device as well as a smart card reader. A driver that supports more than one USB interface is known as a composite driver. A generic composite driver exposes the multiple interfaces of the device to the application software. This is true only if the device has the capability to have more than one interface.

If the device is incapable to support more than one interface due to its architectural limitation, the virtual multi-interface driver according to the invention can be used to overcome the architectural limitation and still expose the device as a multi-interface device. The virtual driver functions just like any other USB composite driver with additional intelligence to handle the multiple interfaces. Thus, with limited hardware, it is possible to get the complete functionality of composite devices.

Figure 7 illustrates several possible application scenarios for the virtual multi-interface driver. The device shown in Figure 7 has a single electrical interface, in particular a mass storage class interface. The virtual multi-interface driver is loaded for the device. It is apparent from the figure that the virtual multi-interface driver reports two logical interfaces to the host. The first logical interface is the mass storage device interface which actually exists in the device; a mass storage driver provided by the operation system of the host (Microsoft Windows, for example) gets loaded for the first interface. The second logical interface is the virtual smart card interface which is created by the virtual multi-interface driver; a smart card driver provided by the operating system of the host gets loaded for the second interface.

The virtual multi-interface driver has the intelligence of projecting a single interface as a composite device to the host. The virtual multi-interface driver achieves this by creating a virtual smart card interface in the driver itself. The virtual interface is a logical one and does not physically exist on the secure digital media reader. The host system accepts that there is a mass storage device and a smart card device present in the system.

But, according to the logic connection diagram shown in Figure 8, the smart card reader shown in Figure 7 is actually a virtual device. Thus, the virtual multi-interface driver successfully emulates a composite device by using a device that is architectured to support only a single interface i.e. only the mass storage interface. A driver letter appears for the mass storage device through which the mass storage device interface can be accessed and the data contents can be read or written from

or to the digital medium. To access the smart card interface, any application which is intended for a valid smart card reader can be used. Both interfaces cannot be accessed simultaneously. When the mass storage device interface is in use the smart card interface is locked and vice versa. But it is possible to switch between these interfaces by giving a single command to the device.

The commands received from the mass storage device driver provided by Microsoft Windows OS are in SCSI command format and are directed to the device as such. This is the function of the mass storage device interface portion of virtual multi-interface driver. The commands received from the smart card driver provided by Microsoft Windows OS are in smart card command format. The virtual multi-interface driver converts smart card command format to SCSI command format and directs the converted commands to the device (see I5 in Figure 7).

Reference is now made to application I2 shown in Figure 7 (digital media with PIN support). The virtual multi-interface driver supports Windows log-on through a digital media that supports PIN. During log-on the user will be prompted to enter a PIN. Once this happens the PIN which the user has entered is compared with the PIN stored in the digital media. If the match is found, the user will be allowed to log on to Windows through this media.

Regarding application I3 shown in Figure 7 (secure digital media reader), the virtual multi-interface driver supports a secure digital media reader according to the invention. The user who wants to access the contents of the digital media should correctly enter the key stored in the smart card. The secure digital media reader thus avoids tampering of critical data stored on the digital medium.

Regarding application I4 shown in Figure 7 (digital media with smart card controller) the virtual multi-interface driver supports access to the digital media with an embedded smart card controller. The device, which is a mediator between the driver and the media, needs to support only a single electrical interface. Since the virtual multi-interface driver has the intelligence of creating virtual logical

interfaces, both mass storage commands and smart card commands received from the host can be handled perfectly. This application of the virtual multi-interface driver gives the user a "look and feel" of using both a smart card reader as well as mass storage reader.

The above-described application scenarios show that the virtual multi-interface driver is not only capable of supporting a device according to the invention in order to read digital media contents which are at least partially secured by a smart card, but also provides backward compatibility for existing media.

Figure 8 further illustrates the software architecture of the system according to the preferred embodiment of the invention. The virtual multi-interface driver is actually a composite driver above which two separate functional drivers (upper interface specific software layers, which are normally shipped with the operating system) get loaded, one for each interface. If there are more than two interfaces provided by virtual multi-interface driver, then so many functional drivers will get loaded above the virtual multi-interface driver. The requests from the application layer are routed to the upper interface specific software layers. These requests are sent to the virtual multi-interface driver. The virtual multi-interface driver just routes the commands to the device and helps to maintain synchronization with the application. Using the operating system provided drivers helps to maintain the application level compatibility.

Figure 10 shows the self-explanatory command flow for the secure digital media reader, with a secure authentication module (SAM) being provided by the smart card.

It has to be understood that the above detailed description refers to a preferred embodiment of the invention. However, the invention is not limited to this embodiment as there are various other embodiments possible within the scope of the accompanying claims which are apparent to a person skilled in the art. For example, the digital media reader may be a device capable of accessing digital media contents from one of the following data sources: a hard disk, a removable

disk, a CD, a DVD, a flash memory, the internet. Further, instead of a smart card reader, any reader capable of reading and transmitting an authentication information may be used, like a reader capable of retrieving biometric information from a user, e.g. a reader including a fingerprint sensor, or an iris, face or voice recognition means.